LAMP-TR-106
CAR-TR-991
CS-TR-4529
UMIACS-TR-2003-97

September 10, 2003

PARSING AND TAGGING OF BINLINGUAL DICTIONARY

HUANFENG MA
BURCU KARAGOL-AYAN
DAVID DOERMANN
DOUG OARD
JIANQIANG WANG

Report Documentation Page

Form Approved
OMB No. 0704-0188

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
2003		**00-00-2003 to 00-00-2003**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Parsing ANS Tagging of Binlindual Dictionary	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of Maryland,Institute for Advanced Computer Studies,Language and Media Processing Laboratory,College Park,MD,20742**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**36**	

LAMP-TR-106 UMIACS-TR-2003-97
CAR-TR-991
CS-TR-4529

PARSING ANS TAGGING OF BINLINDUAL DICTIONARY

HUANFENG MA[1,2]
BURCU KARAGOL-AYAN[1,2]
DAVID DOERMANN[1,2]
DOUG OARD[2,3]
JIANQIANG WANG[2,3]

[1]Language and Media Processing Laboratory
[2]Institute for Advanced Computer Studies
[3]College of Information Studies
University of Maryland, College Park, MD 20742-3275
{*hfma,burcu,doermann*}*@umiacs.umd.edu*
{*oard,wangjq*}*@glue.umd.edu*

Abstract

Bilingual dictionaries hold great potential as a source of lexical resources for training and testing automated systems for optical character recognition, machine translation, and cross-language information retrieval. In this paper, we describe a system for extracting term lexicons from printed bilingual dictionaries. Our work was divided into three phases - dictionary segmentation, entry tagging, and generation. In segmentation, pages are divided into logical entries based on structural features learned from selected examples. The extracted entries are associated with functional labels and passed to a tagging module which associates linguistic labels with each word or phrase in the entry. The output of the system is a structure that represents the entries from the dictionary. We have used this approach to parse a variety of dictionaries with both Latin and non-Latin alphabets, and demonstrate the results of term lexicon generation for retrieval from a collection of French news stories using English queries.

Keywords: Cross-Language IR, OCR, Logical Analysis, Page Segmentation, Bilingual Dictionaries

The support of this research under DARPA cooperative agreement N660010028910 and DOD contract MDA90402C0406 is gratefully acknowledged.

1 Introduction

1.1 The Problem

In recent years, the demand for tools capable searching written and spoken sources of multilingual information has increased tremendously. The involvement of multinational groups in activities all over the world, the proliferation of information on the Internet, and the general explosion of information available in multiple languages have made cross-language communication essential to many organizations. Typically, access to large collections of information in another language requires either the assistance of a speaker of that language to formulate queries and translate the retrieved documents, or an automated system for Cross-Language Information Retrieval (CLIR) and Machine Translation (MT). The former is clearly not practical as a general approach for very large collections, but automated systems for CLIR and MT are rapidly evolving. CLIR systems can, however, often produce acceptable results by performing term weight translation between the query and the document languages, thus allowing the use of well developed term-based search techniques to identify relevant documents. The key enabler for this strategy is to enable translation at the term level by building lexical resources of term-term translation pairs.

We can acquire the needed lexical resources in three ways. Many languages now have a substantial presence on the World Wide Web, and Resnik has shown that substantial amounts of translation-equivalent documents can be found for many languages and that a translation lexicon can be constructed from such a collection using automatic techniques for term-level alignment [33]. As the Web grows, this technique could be extended to an increasingly large set of languages. Similar corpus-based techniques can also be used with documents that are available in printed form [13]. Corpus-based approaches are particularly useful because the learned term-term mappings have associated translation probabilities, but infrequent terms (which are highly valued by retrieval systems because of their selectivity) are rarely observed in the training data and thus rarely included in the learned translation pairs. Hand-built translation lexicons have complementary strengths and weaknesses; they usually lack translation probabilities, but they typically have far

better coverage of rare terms that searchers actually use when posing queries.

Sometimes, these translation lexicons are available directly in electronic form. For example, the translation lexicon in an MT system could be used directly for this purpose. Available MT systems cover fewer than fifty of the world's thousands of languages, however. Online dictionaries are another possible source for this information, but again the number of languages served by such resources is limited. When digital resources are unavailable, printed dictionaries offer the third source of term-term translations. Moreover, bilingual dictionaries also often contain a wealth of supplemental information about morphology, part of speech, and examples of usage that can also be useful in CLIR and MT applications.

The focus of the work reported in this article is on rapid and reliable acquisition of translation lexicons from printed bilingual dictionaries, with an ultimate goal of supporting the development of CLIR systems for languages in which other translation resources are lacking. Given a bilingual dictionary, with one of the two languages English, a scanner, an optical character recognition (OCR) system that is capable of processing the character set, and an operator familiar with the language in question, we can train a retrieval system for the new language in as little as 48 hours. To do this, we have developed an automated, but user guided approach, to parameterize and learn the physical structure of the document page and the semantics of the dictionary entries.

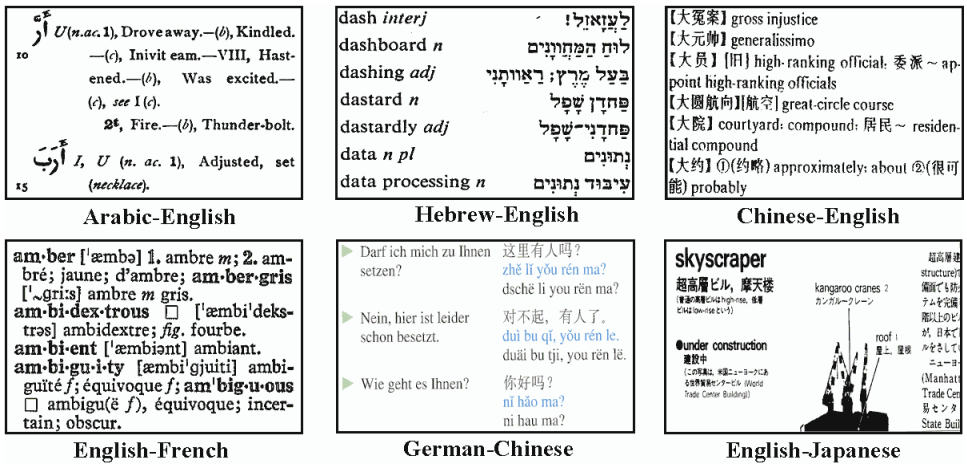

Figure 1: Examples of bilingual dictionaries.

1.2 Background and Approach

Dictionaries are members of a class of documents that are designed for easy search [8]. Their structure is typically regular and repeating, and "keys" are distinguished as access points for each entry. Figure 1 shows some common formats for bilingual dictionaries. The format varies from simple word-to-phrase translation pairs through full descriptions that contain parts of speech, related forms, and examples of usage. Our goal is to capture the salient structure of these entries and label each element appropriately. Because of the regular structure, we are typically able to provide a relatively small number of training samples for each dictionary, and then have the system learn the features necessary for correct segmentation and labeling.

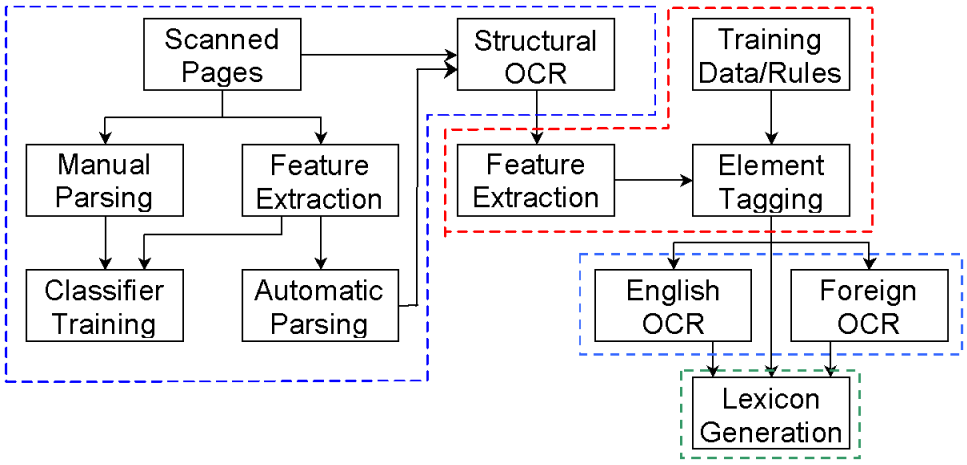

Figure 2: System architecture.

A prototypical architecture for a system is shown in Figure 2. The system is broken up into three main components: Dictionary Parsing, Element Tagging, and Lexicon Generation. The general philosophy we take with all of the components is to provide a trainable system that can learn from a limited number of examples. An operator will identify and tag several dictionary entries, and it is the system's responsibility to then learn the parameters necessary to extract and tag the remainder of the dictionary. Because of the repetitive nature of these documents, a bootstrapping approach where the system gets some feedback from an operator is appropriate. It may ultimately be possible to design general document image analysis systems that are capable of learning

this analysis with unsupervised (or very minimally supervised) techniques, but for now it remains essential to have an operator who can at least understand both languages in the bilingual dictionary. We provide an intuitive interface to assist in processing the dictionary and compilation of results. The use of the resulting translation pairs by a CLIR system can be completely automated, so the overall system development effort is very highly leveraged, requiring only a small amount of language-specific human effort.

1.3 Related Work

Work relevant to entry segmentation is typically found in the logical layout analysis literature ([17] gives an overview of traditional methods), and is often divided into two tasks: physical segmentation into zones and logical labeling. The structural complexity of different documents makes it difficult to design a generic document analysis tool that can be applied to all documents.

In our system, it is essential that we are to learn a dictionary's structure, since that structure is typically consistent throughout a given dictionary but varies widely between dictionaries. Liang et al. [19] presented a probability-based text-line identification and segmentation approach. Their approach consists of two phases: offline statistical training and online text-line segmentation. Kopec et al. [15] applied a stochastic approach to build Markov source models for text-line segmentation under the assumption that a symbol template is given and the zone (or text columns) had been previously extracted. Under the assumption that the physical layout structures of document images can be modeled by a stochastic regular grammar, Mao et al. [24] built a generative stochastic document model to model a Chinese-English dictionary page. They use a weighted finite state automaton to model the projection profile at each level of the document's physical layout tree, and to segment the dictionary page on all levels. Lee et al. [18] proposed a parameter-free method to segment document images with various font sizes, text-line spacing and document layout structures. There are also some rule-based segmentation methods which perform the segmentation based on rules that are either manually set up by the user [17] or learned automatically by training [22, 15].

Although numerous techniques have been proposed for both script identification [10,

35, 34, 39] and font recognition [23, 14, 36, 44, 43], none focused on either word level classification or on optimizing performance for a given document. For entry-level tagging, we attach semantic labels to words or phrases. Palowitch and Stewart [29] constructed a system for automatically identifying structural features in text captured from printed documents via OCR and applying Standard Generalized Markup Language (SGML) tagging, based on the Text Encoding Initiative (TEI) Document Type Definition. They implemented the structural and presentation markup phases as an iterative process.

For tagging, Mao and Kanungo [24] used stochastic language models to automatically extract bilingual dictionary entries by manually creating a context-free grammar (CFG) and stochastic production rules for each dictionary. They demonstrated their algorithm on a Chinese-English dictionary which contained four categories. In the general case, however, manual grammar-based approaches cannot adequately accommodate the uncertainty introduced by OCR and by errors in the document analysis. Wilms [41] reports on an attempt to impose structure on a particular dictionary using a hand-built parser, adopting a mixed approach based on pattern matching and transition networks. After extensive hand-tuning, the resulting translation lexicon was 95% accurate, but correcting the approaches which require that degree of hand tuning would not be suitable for the rapid application development process that we have in mind.

The remainder of this paper is organized as follows. In Sections 2 and 3, we describe the dictionary parsing and tagging processes in detail, along with examples and experimental results. We outline the implementation for the system in Section 5, and we explain how the resulting translation pairs are used to perform cross-language retrieval (with further experiment results) in Section 6. We conclude the paper with a discussion of what we have learned and a few words about our future plans.

2 Dictionary Parsing

The first step in our system involves physically acquiring the page images, segmenting the images into individual entries, extracting functional characteristics of various elements of the entry, and labeling the types of extracted entries.

2.1 Dictionary Acquisition

In our environment, we assume that we have obtained a copy of a printed bilingual dictionary that can be scanned, either page by page or automatically (after busting the binding). Clearly, the quality of the resulting image can significantly effect our ability to analyze the dictionary, so care must be taken to ensure quality. Images are stored uncompressed at 300dpi, and adjustments are made to accommodate paper thickness and print intensity in order to optimize contrast, minimize bleed through, and preserve subtle differences in font properties.

2.2 Entry Segmentation

Entry segmentation addresses the general problem of identifying repeating structures by learning the physical and semantic features that characterize them. Unlike traditional page segmentation problems where zones are characterized by spatial proximity, we often find that publishers of documents with multiple entries (dictionaries, phone books, and other lists) use different font properties (bold, italics, size etc) and layout features (indentation, bullets, etc.) to indicate a new entry. Although such characteristics vary for different documents, they are often consistent within a single document, and hence the task suggests learning techniques.

2.2.1 Overview

The goal of entry segmentation is to segment each page into multiple (sometimes partial) entries or alternatively to organize multiple lines of text as a single entry. Since the extraction of text lines in dictionaries is relatively straightforward, the problem of entry segmentation can be posed as the problem of finding the first (or last) line of each entry.

A significant contribution to the effectiveness of entry segmentation results from application of a bootstrap technique for the generation of new training samples. Bootstrapping helps to make the segmentation adaptive, progressively improving segmentation accuracy. Figure 3 illustrates the approach, with each iteration consisting of:

- **Feature Extraction and Analysis:** extraction of physical characteristics which

indicate an entry boundary.

- **Training and Segmentation:** learning the parameters of an entry segmentation model.

- **Correction and Bootstrapping:** feedback from an operator, who makes corrections to a small subset of the results that contain errors. Using the corrected segmentation results, bootstrapping samples are generated and used to retrain the system.

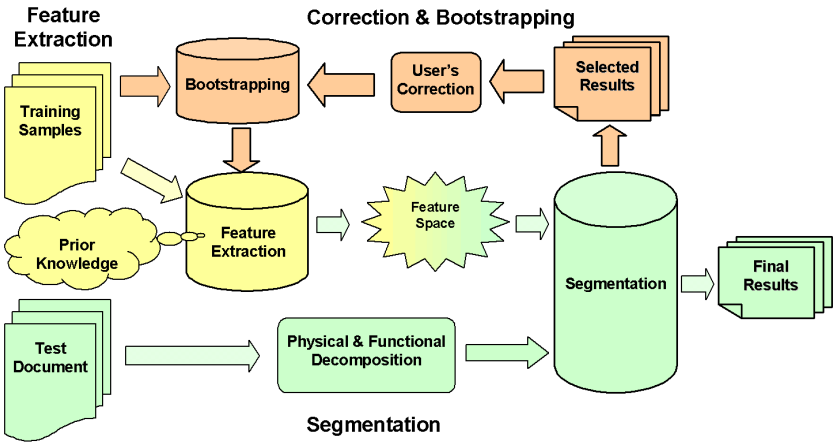

Figure 3: Entry segmentation system architecture.

2.2.2 Feature Extraction and Analysis

Based on a study of different types of structured documents, the following features have been found to be useful for both segmentation and, ultimately, for tagging. Examples are shown in Figure 4.

Special symbols: Punctuation, numbers, and other nonalphabetic symbols are often used to start a new entry, to end an entry, or to mark the continuation of a text line.

Word font, face and size: Word font, face and size (especially the features of the first word in each entry) are often important entry features. On a dictionary page,

for example, the first word of each entry (typically the headword) can be bold, all upper case, a different font, or larger than the rest of the entry.

Word patterns: Words often form distinguishable patterns which are used consistently to convey the structure of an entry.

Symbol patterns: Sequences of symbols sometimes form consistent patterns that represent the beginning or ending of an entry.

Line structures: The indentation, spacing, length, or height of text lines in an entry are represented within our system as "line structures" that may have different characteristics at the beginning or end of an entry.

Other features: Additional features such as regularities in the spacing between adjacent entries, text position on the page, script type, word spacing, and character case may help with interpretation of the principal features described above.

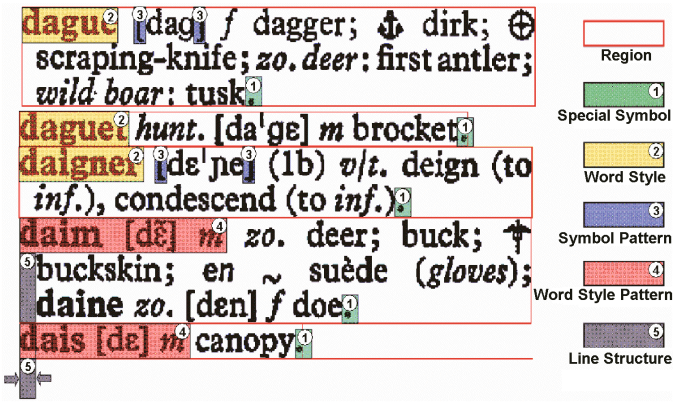

Figure 4: Examples of features.

Features	Weight
First Word Symbol	16.04
Word Style Pattern	10.3
Ending Symbol	15.46
Symbol Pattern	5.77
Word Symbol Pattern	5.77
Line Structure	16.67

Table 1: Features & weights of dictionary in Figure 4.

During the training phase, a Bayesian framework is used to assign and update the probabilities of extracted features. Based on estimated probabilities, each extracted feature

will be assigned a weight that can be used to compute the entry score from all extracted features. The detailed procedure is as follows:

(1) Construct a histogram of feature occurrences in the training samples;

(2) Compute the feature occurrence rate as the empirical feature probability. Suppose there are N training entries, and there are K extracted features. Then for feature i ($1 \leq i \leq K$), the probability can be computed as: $p_i = K_i/N$, where K_i is the number of occurrence of feature i;

(3) Assign feature weights based on this empirical probability as follows: $w_i = \frac{p_i}{A} \times 100$, where $A = \sum_{i=1}^{K} p_i$ and $1 \leq i \leq K$. Consider the extracted features as a formed feature space, each entry is projected to this space and a "voting score" is computed as: $FV = \sum_{i=1}^{K} w_i S_i$, where $S_i = 1$ if feature i appears, otherwise $S_i = 0$;

(4) Obtain the minimum, maximum, and average voting scores of entries; these values will be used as thresholds in the segmentation stage.

Table 1 shows an example of extracted features and assigned weights, where line structure (with heaviest weight) is the most important feature.

2.2.3 Training and Segmentation

Because of the complexity of many structured documents, it would be difficult to manually determine the optimal value for some parameters. So, given a small training set, we attempt to learn appropriate parameters. One way to do that is to extract all possible features from the training set, segment some pages based on the learned features, and generate a new training set from the original set plus selected initial segmentation results. This technique used to generate new "bootstrap" samples was first proposed by Efron [9].

Segmentation is an iterative procedure that maximizes the feature voting score of an entry (computed using the formula described previously) in the feature space. Based on the features extracted in the feature extraction phase, a document can, in principle, be segmented into entries by searching for the beginning and ending text-lines of an entry.

This search operation is a threshold-based iterative and relaxation-like procedure, and the threshold is estimated from the training set. The segmentation procedure can be performed as follows:

(1) Search candidate entries for the first text line in one zone by feature matching. This operation is equivalent to determining whether the first line in one zone is the beginning of a new entry or a continuation of an entry from the previous zone or previous page.

(2) Search for the end of an entry. This operation is actually accomplished by searching for the beginning of the next entry, since the beginning of a new entry ends the previous entry.

(3) Remove the extracted entries, and iterate until all new entries are identified.

2.2.4 Correction and Bootstrapping

In order to correct a system-hypothesized segmentation, the operator can performing one or more operations on any segment: **split** one segment into two or more individual segments, **merge** two or more adjacent segments into a single segment, **resize** a segment to include or exclude text-lines or columns, **move** or change the position of a segment (equivalent to resizing both the beginning and end of a segment, retraining the same number of text-lines), **remove** a segment or **relabel** a segment.

We applied the procedure described in [20] to generate bootstrap samples in such a way that no entry is selected more than once. Generated bootstrap samples are the linear combination of the training samples in source, based on random weights. The result is a set of dictionary entries that represent a physical partition of the page. A similar training and labeling process is applied to each entry on the page to label, for example, entries which are a continuation from a previous page or column. Optical Character Recognition is also applied.

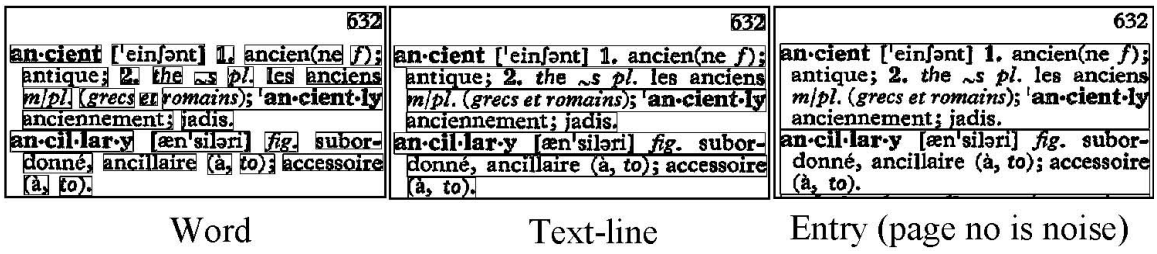

Figure 5: Entry segmentation results, English-French dictionary.

Document	Page No	Total Entries	Correct Entries	Incorrect Entries				False Alarm	Mislabeled Entries
				Missed	Overlapped	Merged	Split		
EnglishFrench	635	20174	**96.11%**	0.005%	1.00%	2.62%	0.26%	0.21%	0.80%
FrenchEnglish	75	2423	**97.90%**	0.04%	1.61%	0.25%	0.21%	0.25%	0.49%
EnglishTurkish	96	3517	**99.26%**	0.00%	0.17%	0.11%	0.45%	0.23%	0.31%
TurkishEnglish	70	2654	**98.98%**	0.04%	0.26%	0.00%	0.72%	0.08%	0.38%
CebuanoEnglish	50	2152	**99.21%**	0.00%	0.14%	0.56%	0.00%	0.00%	4.46%

Table 2: Segmentation results.

2.2.5 Experimental Results of Segmentation

Our system have been developed and implemented in C++ with a Java-based interface. In this section, we highlight entry segmentation results. The segmentation approach was applied to five dictionaries with different structural characteristics: French-English (613 pages), English-French (657 pages), Turkish-English (909 pages), English-Turkish (1152 pages), and Cebuano-English (303 pages). Figure 5 shows the results for the English-French dictionary. The performance is shown in Table 2, which was based on the available ground truth for these dictionaries. Figure 6 shows the performance improvement that results from bootstrapping. The evaluation is performed on 50 pages from each dictionary. The initial segmentation was based on only four entry samples. Iterations following the initial segmentation added different numbers of training entries to generate bootstrap samples. The chart in Figure 6 shows that the segmentation can be refined step by step by applying the bootstrap technique.

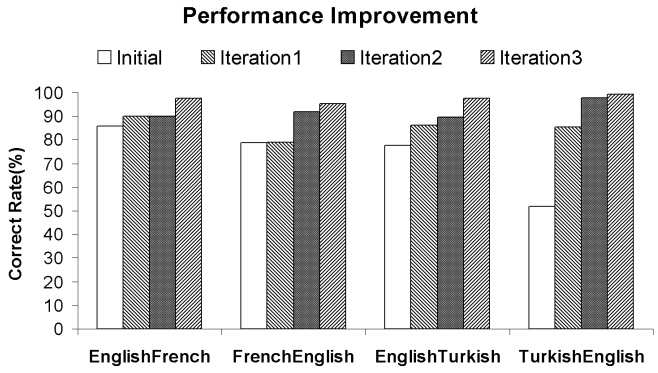

Figure 6: Progressive performance improvement from bootstrapping.

2.3 Functional Labeling

2.3.1 Trainable Classifier Design

Functional properties are properties of words or groups of words that are often used by publishers to implicitly label semantic roles. For example, changes in font-face or font-style are often used to indicate the intended interpretation of specific strings such as pronunciation information, part of speech, or an example of usage. Similarly, the script itself can be considered to be a functional property. Working with documents containing both Latin and non-Latin text requires us to be able to identify scripts before applying an appropriate OCR algorithm. These types of features prove to be extremely useful for both entry recognition and for tagging the constituents of an entry.

Unfortunately, most OCR systems are designed to detect standard variation in these properties. Recognition accuracy depends heavily on the quality of images. Even within the same dictionary, the bold face text may vary significantly among different images, so we use a trainable approach to recognize the font face. Because of the limited number of functional properties, it is very easy for us to provide samples to design an optimal classifier. It should be noted that all the script identification approaches mentioned in the related work are at the block or page level, and the font recognition method in [43] attempts to identify the "primary" font on the page. Obviously, this does not model the case of bilingual dictionaries well, since words in different languages are typically

interspersed; script identification must therefore be done at the word level. In our work, we perform that classification based on a Gabor filter analysis of texture.

The use of Gabor filters to extracting texture features from images is motivated by: (1) the Gabor representation has been shown to be optimal in the sense of minimizing the joint two-dimensional uncertainty in space and frequency [5]; and (2) Gabor filters can be considered as orientation and scale tunable edge and line detectors, and the statistics of these micro-features in a given region are often useful for characterizing the underlying texture information. The system to classify scripts, font-styles and font-faces is shown in Figure 7.

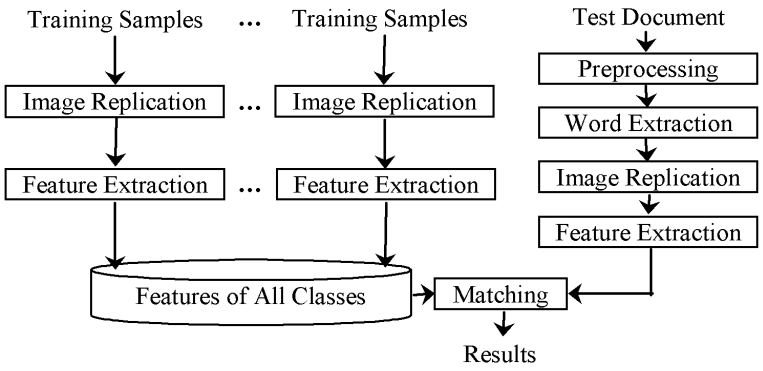

Figure 7: Flowchart of classification system.

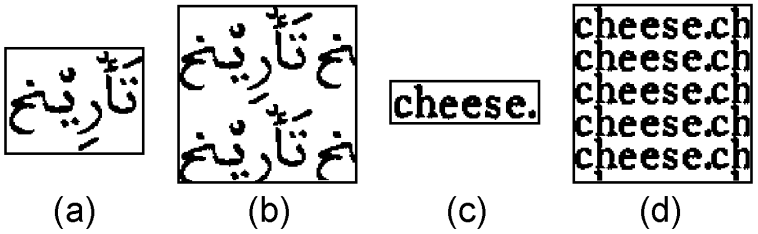

Figure 8: Examples of image replication: (a,c) before; (b,d) after.

We created a set of training samples for each class by randomly picking entries from a large set of words belonging to that class. Words in different scripts, different font-styles, and different font-faces in the same script have different dimensions (width & height), so we use word image replication to generate a new image with predefined size (64x64 pixels in our case), all features are extracted at that image size. For computation, the

Scripts	NNM			SVM		
	R_{avg}	R_{min}	R_{max}	R_{avg}	R_{min}	R_{max}
Arabic/Roman	**73.27%**	67.42%	82.91%	**90.43%**	85.14%	94.17%
Chinese/Roman	**84.08%**	73.36%	89.55%	**90.10%**	82.68%	94.6%
Korean/Roman	**83.49%**	78.98%	87.03%	**90.91%**	86.88%	93.57%
Hindi/Roman	**92.08%**	87.53%	100%	**93.28%**	87.95%	96.73%

Table 3: Script identification performance comparison.

	Normal	**Bold**	*Italic*
Total No	250	40	44
Identified	184	35	39
Accuracy	**73.6%**	**87.5%**	**88.6%**

Table 4: Font-face identification.

	Total No	Times	Arial	Detection
Times	39	39	7	**100%**
Arial	39	7	32	**82.1%**

Table 5: Font-style identification.

replication is to a power of two such that a Fast Fourier Transform can be applied to compute the Gabor filter features. Figure 8 shows word replication examples of two different scripts (Arabic and English).

After generating a word images, multi-channel Gabor filter technique was applied to extract the texture features of each class. We used the same pairs of isotropic Gabor filters as in [43] to extract 16 channels of texture information. We implemented two classifiers, one using Nearest Neighbor Matching (NNM) [21] and one using a support vector machine (SVM) [3].

2.3.2 Results of Functional Labeling

We applied the script identification approach to Arabic-English, Chinese-English, Hindi-English, and Korean-English bilingual dictionaries. Features for each script class were extracted from samples on a single page, and the two classifiers were trained using the same samples. Table 3 shows the performance comparison, where the results were based on 20 (32 for Chinese) pages of identification results. The comparison shows the SVM-based script identification works much better than the NNM-based technique, with the SVM achieving an average accuracy above 90%. The font-face classification approach was applied to the identified English (Roman) document images, and one page of the identification results for the Korean-English dictionary is shown in Table 4. In order to test the effectiveness of this classifier on font-styles, we applied it to an artificial test document image. The classifier was trained by words taken from a different document

Headword (Lexicon)	Translation
Pronunciation	Tense
Part of speech (POS)	Gender
Plural Form	Number
Domain	Context
Cross reference	Language
Antonym	Derived word
Synonym	Derived word translation
Inflected form	Example
Irregular form	Example translation
Alternative spelling	Idiom
Explanation	Idiom translation

Table 6: Some categories typically found in bilingual dictionaries.

(each with 10 training words). Table 5 gives the classification result. Trainable techniques are also applied to recognize special symbols and to correct symbols the OCR fails on.

3 Tagging

The functional segmentation module provides the entry, entry type, list of tokens[1] in the entry, properties (font, font face, script, etc.) of each token, and OCR results. Once we have segmented the dictionary into individual entries, we need to tag each token (or group of tokens) with the role of element in the entry, which we refer to as the "linguistic category." As with entry segmentation, tagging relies on the existence of a repeated semantic structure. Unfortunately, however, the semantic structure of the entries typically varies a great deal more than the physical structure of the entries.

Our approach to tagging is to use (1) the functional properties of tokens, such as font, font face, font style and their relationships to each other, (2) keywords (common terms or symbols, often defined in the preface to the dictionary), and (3) separators (typically punctuation) to tag the entries in a systematic way. The key is to be able to discover the regularities in the occurrences of these clues and make use of them to assign each word to a linguistic category. A list of categories that are commonly found in bilingual dictionaries is shown in Figure 6. Typically, only a subset of those categories is used in a given dictionary, but our system is extensible at runtime to add new categories for additional constructs.

[1]We define a *token* as a set of glyphs in the OCR output that is separated by white space. We refer to a group of one or more tokens as an *element*.

récolte [reˈkɔlt] *f* harvest, crop; harvesting; F *fig.* collection; *fig.* profits *pl.*; **récolter** [ˌkɔlˈte] (1a) *v/t.* harvest; gather in; *fig.* collect. **recommandable** [rəkɔmɑ̃ˈdabl] to be recommended; estimable (*person*); *fig.* advisable; **recommandation** [ˌdaˈsjɔ̃] *f* recommendation; *fig.* instruction, advice; *post*: registration; **recommander** [ˌˈde] (1a) *v/t.* recommend; *fig.* advise; *fig.* bring (*to s.o.'s attention*); *post*: register; se ~ à commend o.s. to; se ~ de give (*s.o.*) as a reference; *post*: en recommandé by registered post (*Am.* mail).

Figure 9: French-English dictionary.

Publishers typically use a combination of methods to indicate the linguistic category. As illustrated in Figure 9, functional properties (changes in font, font style, font-size, etc.) can be used to implicitly indicate the role of an element, keywords can be used to make that role explicit, and various separators can be used to organize them. For instance, headwords may be in bold, examples of usage in italics, the keywords *adj*, *n* or *v* may be used to explicitly represent the part-of-speech, pronunciation may be offset with brackets, commas may be used to separate different examples of usage, and a numbering system may be used to enumerate variant forms.

Since there can be errors in both functional labeling and OCR, we need methods that are robust to local variations. We have therefore developed two complementary approaches. The first approach is rule-based, which is instrumental in understanding the structure of the dictionary (Section 3.1). The second is a Hidden Markov Model (HMM)-based approach that aids in overcoming errors introduced during segmentation (Section 3.2). The overall tagging architecture is shown in Figure 10.

3.1 Rule-Based Approach

For the rule-based approach, an operator uses explicit knowledge of the structure of the dictionary entries to select an inventory of tag types and to indicate how they typically appear in the scanned dictionary. The operator can select from the pre-defined list of

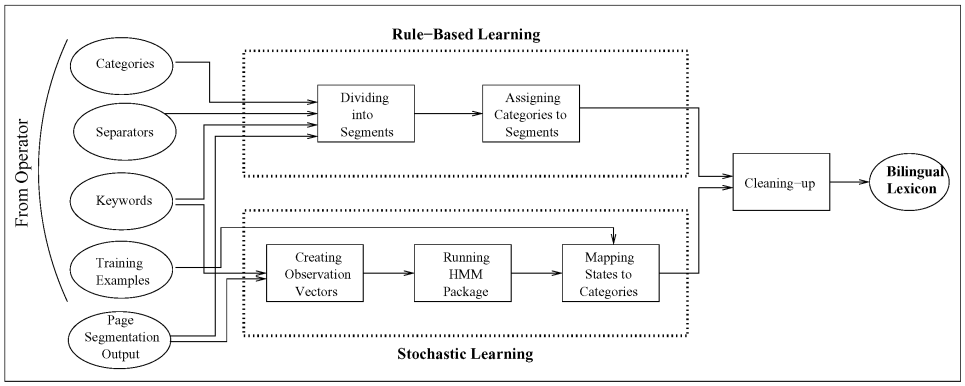

Figure 10: Architecture for entry tagging.

possible categories (linguistic meanings), and if a category is found in the dictionary but not on that list, the new category can be added. Similarly, physical attributes are selected from a list (normal, bold, italics, etc). Keywords are identified in a table provided by the operator, typically based on the legend at the beginning of the dictionary. For each keyword, we associate a single linguistic meaning. Separators (or symbols) and their functions are defined using the operators shown in (Table 7). Typically *InPlaceOf* and *Contains* operands are sufficient, but in some dictionaries, spatial proximity and relative position are important, so additional operators are provided for such cases.

The tagging proceeds as follows. Using the font style and separator as dividers, the tokens in an entry are grouped into elements that will be tagged as a unit. The resulting elements are spans that are (1) rendered in a consistent font style, (2) identified as one of the given keywords, and/or (3) delimited by one of the separators from the given set. In Figure 9, for instance, the tokens *to, be, recommended* are grouped together as a single element since there is no separator between them, the token before them ends with the symbol ']', they end with a semi-colon, and they have the same font. Each group of tokens will be assigned a single linguistic category. Because uncertainty in the document image analysis process leads to errors in the segmentation, several rules can be created for each category. For instance, there are some cases where the separators are recognized incorrectly, so we might choose to indicate the pronunciation begins with either "(" or "[".

After the tokens in an entry are grouped, the tagging process attempts to associate a single tag with each group. In this process, we again make use of font styles, keywords, and separators. We first check to see whether the group we are considering is in the list of keywords and if it is, we tag it accordingly. Otherwise, we use the separators and the font information. Here, we arrange the categories in a precedence order, first checking for translated tokens in the primary language (a derived form of the headword, an example, or an idiom), then the translations, and finally the supplemental categories, such as pronunciation, part-of-speech or gender. This order is tuned to our follow-on task, extracting term-term translation pairs, but it could be adapted for other applications.

Operand	**Definition**	**Example**
¡cat¿ InPlaceOf ¡sym¿	Used as a shortcut for a category	**InPlaceOf** headword ∼
¡cat¿ StartsWith ¡sym¿	Category begins with this separator	pronunciation **StartsWith** [
¡cat¿ EndsWith ¡sym¿	Category ends with this separator	translation **EndsWith** ;
¡cat¿ PreviousEndsWith ¡sym¿	Previous category ends with this separator	translation **PreviousEndsWith** ¡comma¿
¡cat¿ Contains ¡sym¿	Category contains this separator	derived **Contains** ¡dot¿

Table 7: Operators used to model separators (¡cat¿=category, ¡sym¿=symbol).

When applying operators, all must be valid for the category to be assigned to the element. This strict approach makes it practical to rapidly write robust rules without tuning their firing order (e.g., when font is the only feature that distinguishes between one pair of categories). If no category satisfies all of the constraints, operators are tried individually until a match is found. If we still cannot decide the category based solely on the observed features and the available rules we assign a "miscellaneous" tag to the element (this tag can also be assigned by a rule if desired). As the last step of the rule-based tagging process, we apply a cleanup process in which some punctuation and other nonalphabetic characters are removed by default. The operator can also specify additional characters to be cleaned if necessary.

3.2 Statistical Learning Approach

For dictionaries with highly regular structures and fairly accurate OCR, the rule based system effectively leverages human abilities to generalize from a limited number of exam-

ples. Moreover, the understandable nature of the rules facilitates failure analysis, thus allowing rapid iteration to an effective rule set. However, it can be difficult to anticipate the effects of OCR errors, and subtle variations in style can complicate the process of generating an effective set of rules. Because of this we have also developed a stochastic learning method using Hidden Markov Models (HMMs).

HMMs have been successfully used in several applications such as speech recognition [1, 11] and part of speech tagging [16, 40, 26]. An HMM is specified by a five-tuple (S, O, Π, A, B) which corresponds to hidden states, observable states, initial state probabilities, hidden state probabilities, and symbol emission (transition) probabilities [23]. In our tagging process, the observable states are the tokens in the dictionary entry, and the hidden states are the tags for those tokens. Given the observation sequence, we would like to find a parameter set $\lambda = (\Pi, A, B)$ that maximizes the conditional probability, $Pr(O|\lambda)$. In our case, each observable state has a feature vector that consists of six features that represent the clues that can be useful for determining the linguistic category. These features and possible values they can take are as follows:

- *Content*: Category of the keyword if the token is a keyword, *SYM* if it is a special symbol, *NUM* if it is a number, and `null` otherwise.
- *Font*: Font style of the token (normal, bold, italic, etc.).
- *Starting symbol*: Special punctuation mark if the token begins with one, `null` otherwise.
- *Ending symbol*: Special punctuation mark if the last character of the token is one, `null` otherwise.
- *Second ending symbol*: Special punctuation mark if the second to last character of the token is one, `null` otherwise.
- *Is-first token*: True if this is the first token of an entry, false otherwise.

We used an HMM software package developed in our group [6]. For training the HMM parameters, a hybrid method is applied that uses the Baum-Welch algorithm [2] for the

first ten iterations and then a segmental K-means algorithm [12, 31] for the remainder. This hybrid method runs faster than the Baum-Welch approach, and gives comparable results. The Viterbi algorithm [37] is used for decoding (i.e. determining the sequence of hidden states that is most likely to have produced an observation sequence).

Each token in the dictionary must be transformed to an observation vector in the described format before the HMM is run. For that transformation, the list of keywords and their corresponding categories is needed. The final mapping of hidden states to categories is done using a small hand-annotated training sample of 400 randomly selected tokens that are assigned to their correct tags manually. This ground truth is used to find the category that each state represents. As a final step, the cleaning process described in Section 3.1 is applied to HMM output.

3.3 Experiments

The purpose of the experiments performed for entry tagging is to show how well the original dictionary entry structure is captured by the system, independent of other components. Precision, recall and F-measure are calculated based on a labeling of each of the categories described above. Precision (P) measures how accurately we labeled the various elements. It is computed as *Number of correct tags/Number of tags given by the system.* Recall (R) is a measure of coverage. It is computed as *Number of correct tags/Number of tags in the ground truth data.* F-measure (F) is a weighted harmonic mean of precision and recall, which can be computed as $F = (\beta + 1)PR/\beta(P + R)$. We took $\beta = 1$, giving recall and precision equal weight.

We performed entry tagging on the first four dictionaries listed in Section 2.2.5 and evaluated the system by preparing ground truth manually for five selected pages of the French-English dictionary and English-Turkish dictionary. We performed two experiments. The first was token-based, with each token considered individually, even if it is part of a larger element. The second was phrase-based, where we treated a tagging as correct only if the tokens were both segmented correctly into an element and and that element then tagged correctly. For instance, if the correct translation for the word *récolter* is *gather in*, but the system produces two separate translations *gather* and *in*,

then these two would be counted as correct in token-based evaluation, but as incorrect in the phrase-based evaluation.

The results in Table 8 were tabulated for two configurations: one in which all linguistic categories were counted, and a second in which only headwords (or derived words) and their translations were counted. The first of these models the requirements of a full MT system, the second the minimal requirements for CLIR.

French-English Dictionary

		All categories			Hw/Der. Word Trans.		
Learning system	**Evaluation method**	**P**	**R**	**F-m**	**P**	**R**	**F-m**
Rule-based	Token-based	72.89	72.89	72.89	67.55	**77.62**	**72.23**
Rule-based	Phrase-based	74.60	75.88	75.23	65.67	76.93	70.86
Stochastic	Token-based	**77.69**	**77.69**	**77.69**	**70.67**	62.35	66.25
Stochastic	Phrase-based	69.62	72.25	70.91	70.57	60.63	65.22

English-Turkish Dictionary

		All categories			Hw/Der. Word Trans.		
Learning system	**Evaluation method**	**P**	**R**	**F-m**	**P**	**R**	**F-m**
Rule-based	Token-based	86.85	86.85	86.85	**84.75**	87.74	86.22
Rule-based	Phrase-based	**88.84**	**87.96**	**88.40**	83.82	**89.17**	**86.41**
Stochastic	Token-based	87.55	87.55	87.55	80.06	85.72	82.79
Stochastic	Phrase-based	79.43	75.11	77.21	67.33	62.93	65.06

Table 8: Experimental results.

For the French-English dictionary, the stochastic learning method with token-based evaluation gave the highest precision, with 78% precision and recall when all categories are considered. For Headword/Derived word translations, the highest precision was 71% with stochastic learning and token-based evaluation, and the highest recall was 78% with rule-based learning and token-based evaluation. For this dictionary, the precision range was 70%-78% and recall range was 72%-78% across all categories.

English-Turkish dictionary tagging yielded substantially better results. The reason for this is that the dictionary had a simpler structure. The rule-based method with phrase-based evaluation performed better. The best precision for Headword/Derived word translations is 85% with rule-based method and token-based evaluation. For all categories, precision is in the range 79%-89% and recall is in the range 75%-88%. When only Headword/Derived word translations are considered, precision range is 67%-85% and recall range is 63%-89%.

One problem with applying HMMs to different dictionaries is finding the correct

number of states that gives the maximum performance. We tried different number of states with the two dictionaries. The number of states that gives the best performance is not the same for the two dictionaries, and we took the best performance for each one (55 states for French-English and 35 states for the English-Turkish dictionary). As the dictionary structure becomes more complex, it seems that an HMM with more states performs better.

The rule-based method usually performs better under phrase-based evaluation. Only for some Headword/Derived word translations did token-based evaluation perform better than the phrase-based. The rule-based method is better suited for dictionaries with simpler structures as well. Stochastic learning method with phrase-based evaluation gives the lowest precision-recall values for both dictionaries. However, we are considering some post-processing to HMM-results to improve phrase-based evaluation.

Table 9 shows detailed results for each category of the French-English dictionary. The example and example translations are the categories that degrade the performance of the tagging process the most for this particular dictionary. The reason for this is that the only information to find the separation point between an example and its translation is the font, which is less reliably extracted. In Figure 9, for instance, the example usage *en recommandè* and its translation *by registered post* are recognized as normal font and there is no separators between them, therefore there is no way to tell the separation point between them. As a result, the five tokens were grouped together, and tagged as a translation. For tags that are represented with keywords, the mistagging occurs because of spelling errors in the input. The reason that the number of miscellaneous token is so high is that this particular dictionary has many image icons, and we are tagging them as miscellaneous. Ideally, the tagging should be optimized to the resulting application. For example, CLIR should maximize recall to generate a complete term list, while MT will typically benefit from example/translation pairs.

4 Generation

Generation is the process of producing different derivatives of the basic structure that can be used in different tasks. One simple output is the lexicon-translation pair (i.e. bilingual

TAGS	Total	Produced	Correct	Precision	Recall	F-measure
Headword	160	160	157	98	98	98
Pronunciation	272	267	259	97	95	96
POS	115	79	79	100	69	69
Domain	52	47	47	100	90	95
Gender	193	183	178	97	92	95
Number	28	26	25	96	89	93
Headword Translation	396	481	334	69	84	76
Context	65	53	52	98	80	88
Language	9	9	9	100	100	100
Alternative Spelling	32	25	17	68	53	60
Derived Word	115	93	80	86	70	77
Derived Word Translation	271	292	195	67	72	69
Example	117	50	13	26	11	16
Example Translation	130	117	27	23	21	22
Explanation	108	119	107	90	99	94
Abbr.	6	0	0	0	0	0
Misc.	208	314	142	45	68	54

Table 9: Detailed results for each tag of the French-English dictionary using the rule-based approach

récolter *(Compound/Derived)*
Pronuncation:-kDlte
*Explanation:*la
*POS:*v/t [transitive verb]
*Translation:*harvest
*Translation:*gather in
*Context:*fig [figuratively]
*Translation:*collect
recommandable
*Pronuncation:*rak müdabl
*Translation:*to be recommended
*Translation:*estimable
*Explanation:*person
*Gender:*f [feminine]
*Translation:*ig
*Translation:*advisable

récolte	harvest	**recommandable**	ig
récolte	crop	**recommandable**	advisable
récolte	harvesting	**recommandation**	recommendation
récolte	collection	**recommandation**	instruction
récolte	profits	**recommandation**	advice
récolter	harvest	**recommandation**	registration
récolter	gather in	**recommander**	recommend
récolter	collect	**recommander**	advise
recommandable	to be recommended	**recommander**	bring
recommandable	estimable	**recommander**	register

Figure 11: HTML representation and bilingual term list.

term list) used in our CLIR experiments. In general, however, the representation is a hierarchical nesting of elements and their scope. Each translation for example may have a POS, or usage associated with it. Figure 11 shows a portion of the derived representation for the dictionary in Figure 9. Two representations, an HTML-based showing all information in the dictionary and the bilingual term list, are shown for the same dictionary portion. The keyword *fig.* in the original dictionary page was recognized as two tokens, *f* and *ig.*, and they are tagged as a POS and a translation, which is wrong.

We did not remove the duplicate translations for the input to CLIR applications,

since these will not reduce the performance. If the user wants to remove duplicates, this can be done during the clean-up process.

5 System Implementation

We have implemented the entire system, and integrated it with an interface and third party software for OCR (Figure 12). For segmentation, it allows the operator to identify entries on the image for training, trains the system and facilitates manual correction and bootstrapping, as well as providing tools for evaluation. For tagging, a customizable interface allows the operator to tailor the configuration for a given dictionary, selecting valid categories, entering keywords and defining separators. Training data is tagged with pull down menus configured in the category list. When the system is run, the results are shown as color-coded boxes overlaid on the original image. The results of generation of term lexicons are shown on a separate panel.

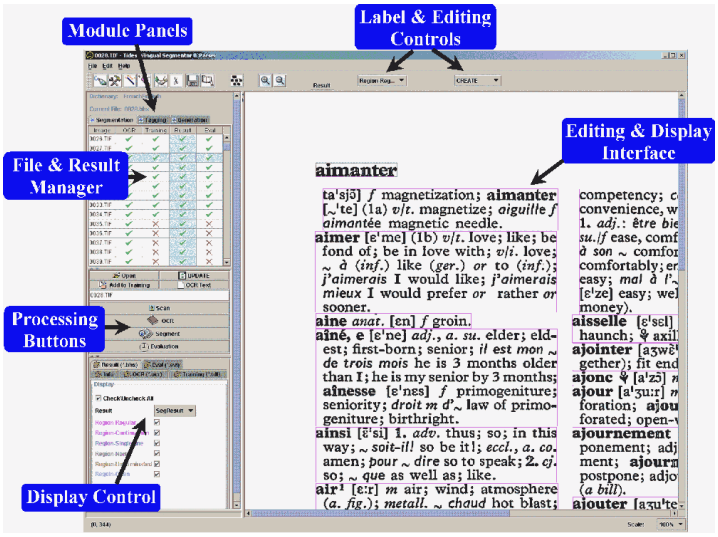

Figure 12: System implementation.

6 Cross Language Retrieval Applications

The key goal of a cross-language information retrieval system is to identify promising documents, even if those documents are not written in the language in which the query

is expressed. As with typical Web search engines, the usual approach is to accept a query from the user and then return a ranked list of documents that the user might wish to examine. There are two basic approaches to the cross-language aspect of this task: translating the queries into the document language or translating the documents into the query language [27]. Document translation has advantages in high-volume applications with a single query language; query translation achieves greater flexibility, but at the expense of somewhat slower processing of each query. We chose a query translation architecture for our experiments because that allowed us to try several types of translation resources without reindexing the collection. Figure 13 shows the major steps in our retrieval experiment.

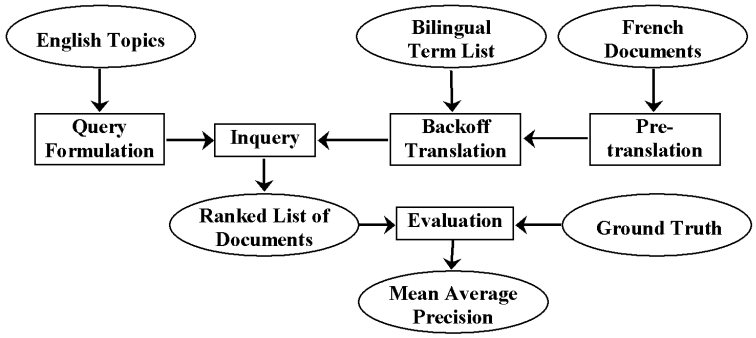

Figure 13: Test system architecture.

We searched a set of 44,013 French news stories from 1994 editions of Le Monde, formulating queries from 40 written topic descriptions from the Cross-Language Evaluation Forum (CLEF), a standard information retrieval test collection created for evaluations in Europe during 2000 [28]. The topic descriptions express the information need in a few words (referred to as the "title"), in a sentence (description), and in a very brief paragraph (narrative). We formed queries of three lengths: title ("T," which is designed to be representative of the very terse queries often posed by Web searchers), title and description ("TD," similar to a brief statement of interest that might initially be given to a reference librarian), and title, description and narrative ("TDN," a fairly complete description of what the user has in mind). CLEF topic descriptions are available in English and French; we used the English topics for cross-language retrieval and the French topics

for a contrastive monolingual condition. This resulted in three sets of 40 French queries each and three corresponding sets of 40 English queries. As a measure of retrieval effectiveness, we report mean uninterpolated average precision as the expected value (over the topics) of the average (over the set of relevant document for each topic) of the fraction of documents at or above that relevant document that are relevant. This models the case in which a searcher poses a query that is typical of those in the evaluation set and then chooses to peruse the resulting ranked list down from the top until they have seen some desired number of relevant documents [38]. Averaging over an ensemble of queries helps to prevent undue preference for systems that perform well only on a narrow range of query types (e.g., those that contain proper names).

We use a two-tailed paired t-test, and report statistical significance for p¡0.05. Pirkola's method was used for structured query translation, which has been shown to perform well when translation probability information is not available [30]. We used a simple rule-based system to split clitics in the French documents and then stemmed the resulting terms. We used a statistical stemmer to remove endings that were unexpectedly common, given the observed character co-occurrence statistics of the language [28]. We then stripped all accents and built a single index of the resulting terms that was used in all of our experiments.

We embedded query translation in a similar processing stream. Specifically, we automatically performed list-based English stopword removal, query translation into French, French clitic splitting, French stemming, and accent removal, in that order. For query translation, we used the term occurrence statistics for every known French translation of an English query term to compute the statistics for that query term. If no translations were known, we instead used the term occurrence statistics for all known translations of any English term that shared a common stem with the query term (using the rule-based Porter stemmer for English). In earlier work, we have found that this backoff translation technique achieves a good balance between accuracy (precision) and comprehensive coverage (recall) [32].

As lower and upper baselines for cross-language retrieval effectiveness, we used wrong-language and same-language retrieval, respectively. For wrong-language retrieval, we

omitted the translation stage, and used English queries to search French documents. This actually works quite well for some queries, since our normalization process results in some matches on proper names and other cognates. When averaged over a large set of topics, the retrieval effectiveness of such an approach is poor. This therefore provides a reasonable lower baseline; if we can't beat wrong-language retrieval, our translation technique is not adding value. For same-language retrieval, we use queries formed from the CLEF French topic descriptions and again omit the translation stage. Since the CLEF French and English topic descriptions are related by human translation (in one direction or the other), this provides a reasonable approximate upper bound on cross-language retrieval effectiveness. The upper bound is only approximate, however, since this approach does not model the sometimes-beneficial synonym-expansion effects that are inherent in Pirkola's structured query translation method.

We obtained the set of known translations for our main experiments by merging translation lists from the English-to-French and French-to-English portions of our dictionary into a single bilingual term list containing 145,247 unique translation pairs. We chose the French-English for our experiments in part because good translation resources exist for that language pair in electronic form. We have used two such resources to establish reference points to which our results from the scanned bilingual dictionary can be compared. The most directly comparable translation resource that we used was a manually created clean bilingual term list obtained from the Internet. That bilingual term list contains 30,322 unique translation pairs, which several studies have shown to be enough to approach a point of diminishing returns in cross-language retrieval applications [7].

This bilingual term list thus provides a somewhat tighter upper bound than same-language retrieval on the retrieval effectiveness that we could reasonably expect from our query translation architecture if our zoning, OCR and segmentation accuracy were not limiting factors. Finally, we have previously demonstrated that bilingual term lists automatically inferred from sets of translation-equivalent Web pages identified by the STRAND system can also be used as a basis for cross-language information retrieval[32]. We therefore also repeated our experiments with a bilingual term list that we had made in that way.

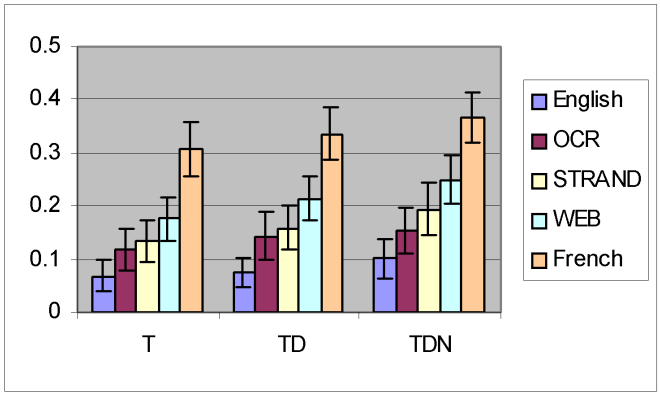

Figure 14: Mean average precision and associated 95% confidence intervals for 15 CLEF runs.

Figure 14 shows the mean average precision and 95% error bars for the experiments using 3 types of queries (title, title+description, title+description+narrative) and 5 different versions of each: English queries (ENGLISH), French queries (FRENCH), and translated English queries using the lexicon extracted from the bilingual dictionaries (OCR), from parallel text (STRAND) and from the online dictionary (WEB). Imperfect zoning, OCR and parsing clearly have an adverse effect on retrieval effectiveness when compared to use of a clean bilingual term list. The differences between OCR/WEB and OCR/STRAND are statistically significant for all three query lengths. Of course STRAND and WEB resources may not always be available. Our results suggest that integrated techniques that draw on both sources of evidence might be beneficial for language pairs with a substantial Web presence.

7 Discussion and Conclusion

While the application we have presented in this paper is fairly unique, the underlying techniques used to solve the problem have great generality. In document analysis, the problem of bootstrapping a system to rapidly be able to provide segmentation and OCR for structured documents is of great interest for high volume applications and in the field of interactive document image analysis. The parsing and labeling of structured text is a problem that has applications to interpretation of email, forms and business

correspondence. Furthermore, the problem opens up a wide variety of questions about how we deal with problems like cross language retrieval in the presence of noisy resources. As a summary to this paper, we will outline several challenges that lie ahead, and some of the outstanding research issues this work motivates.

- OCR Error Modeling: At present, when we fail to find any translations for a query term, we are using any known translations of morphological variants of the term. An obvious extension to this idea is to use any known translation of the terms that are related to the query term though plausible OCR errors. We have developed a fairly sophisticated error model and our next step will be to integrate them in to our translation process.

- Multiple dictionaries: It is now widely agreed that incorporating evidence of translation probability can improve cross-language retrieval effectiveness [42]. Some printed bilingual dictionaries list translations in preference order, and we have previously demonstrated techniques for inferring translation probability from the pattern of presence or absence of translations in multiple dictionaries [4]. Perhaps even more importantly, we expect results from multiple dictionaries to exhibit a degree of independence, suggesting that merging results from multiple dictionaries can help to overcome limited extraction recall in any single dictionary.

- Query expansion: Pre-translation query expansion has been shown to be an effective way of overcoming limitations in bilingual term list coverage [25]. Both knowledge-based techniques (e.g., thesaurus-based expansion) and statistical techniques (e.g., blind relevance feedback) can be used for this purpose, although considerable tuning is often required to obtain optimal results. Since our focus is on supporting English queries, it seems likely that the same tuning effort can be leveraged over multiple language pairs. Post-translation query expansion single words to either single words or multi-word expressions. Reversing the term list produces some mappings from multi-word expressions, which can be beneficial since multi-word expressions typically exhibit little translation ambiguity. Our present translation techniques are word-based, however, (as a consequence of our approach to pre-translation stop-

word removal), so we fail to exploit this potential. Moreover, special processing is required to accommodate multi-word expressions in the document language with Pirkola's method, and our handling of that case is not optimal. Both can and will be improved.

References

[1] J. K. Baker. The dragon system – an overview. In *IEEE Transactions on Acoustics, Speech, and Signal Processing*, volume 23, pages 24–29, 1975.

[2] L. E. Baum. An inequality and associated maximization technique in statistical estimation for probabilistic functions of markov processes. In *Inequalities III: Proceedings of the Third Symposium on Inequalities*, pages 1–8, University of California, Los Angeles, 1972. Academic Press.

[3] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[4] K. Darwish and D. W. Oard. Clir experiments at maryland for trec 2002: Evidence combination for arabic-english retrieval. In *The Eleventh Text Retrieval Conference (TREC 2002)*, 2002.

[5] J. G. Daugman. Complete discrete 2d gabor transforms by neural networks for image analysis and compression. *IEEE Trans. Acoustics, Speech and Signal Processing*, 36:1169–1179, 1988.

[6] Daniel DeMenthon and Marc Vuilleumier. Lamp_hmm v.0.9. http://www.cfar.umd.edu/∼daniel/LAMP_HMM.zip, 2003. software.

[7] D. Demner-Fushman and D. W. Oard. The effect of bilingual term list size on dictionary-based cross-language information retrieval. In *to appear on 36th Hawaii International Conference on System Sciences*, Hawaii, 2003.

[8] D. Doermann, E. Rivlin, and A. Rosenfeld. The function of documents. *International Journal of Computer Vision*, 16(11):799–814, 1998.

[9] B. Efron. Bootstrap methods: Another look at the jackknife. *Annual Statistics*, 7:1–26, 1979.

[10] J. Hochberg, P. Kelly, T. Thomas, and L. Kerns. Automatic script identification from document images using cluster-based templates. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(2):176–181, 1997.

[11] F. Jelinek, R. L. Mercer, and L. R. Bahl. Design of a linguistic statistical decoder for the recognition of continuous speech. In *IEEE Transactions on Information Theory*, volume 21(3), pages 250–256, 1975.

[12] B. H. Juang and L. R. Rabiner. The segmental k-means algorithn for estimating parameters of hidden markov models. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(9):1639–1641, 1990.

[13] T. Kanungo and P. Resnik. The bible, truth, and multilingual ocr evaluation. In *SPIE Conference on Document Recognition and Retrieval*, pages 86–96, 1999.

[14] E. Kavallieratou, N. Fakotakis, and G. Kokkinakis. Slant estimation algorithm for ocr system. *Pattern Recognition*, 34(12):2515–2522, 2001.

[15] G. E. Kopec and P. A. Chou. Document image decoding using markov source models. *IEEE Tran. Pattern Analysis and Machine Intelligence*, 16(6):602–617, 1994.

[16] J. Kupiec. Robust part-of-speech tagging using a hidden markov model. In *Computer Speech and Language*, volume 6, pages 225–242, 1992.

[17] F. LeBourgeois, S. Souafi-Bensafi, and J. Duong. Using statistical models in document images understanding. In *DLIA2001 Advance Program*, Seattle, WA, 2001.

[18] S. Lee and D. Ryu. Parameter-free geometric document layout analysis. *IEEE Tran. Pattern Analysis and Machine Intelligence*, 23(11):1240–1256, 2001.

[19] J. Liang, I. T. Phillips, and R. M. Haralick. An optimization methodology for document structure extraction on latin character documents. *IEEE Tran. Pattern Analysis and Machine Intelligence*, 23(7):719–734, 2001.

[20] Huanfeng Ma and David Doermann. Bootstrapping structured page segmentation. In *SPIE Conference Document Recognition and Retrieval*, pages 179–188, Santa Clara, CA, 2003.

[21] Huanfeng Ma and David Doermann. Gabor filter based multi-class classifier for scanned document images. In *7th International Conference on Document Analysis and Recognition*, pages 968–972, Edinburgh, Scotland, 2003.

[22] D. Malerba and F. Esposito. Learning rules for layout analysis correction. In *DLIA 2001 Advance Program*, Seattle, WA, 2001.

[23] Christopher D. Manning and Hinrich Schutze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[24] Song Mao and Tapas Kanungo. Stochastic language models for automatic acquisition of lexicons from printed bilingual dictionaries. In *Document Layout Interpretation and Its Applications*, Seattle, WA, 2001.

[25] Paul McNamee and James Mayfield. Comparing cross-language query expansion techniques by degrading translation resources. In *25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 159–166, Tampere, Finland, 2002.

[26] B. Merialdo. Tagging english text with a probabilistic model. *Computational Linguistics*, 20(2):155–172, 1994.

[27] Douglas W. Oard and Anne R. Diekema. Cross-language information retrieval. In *Annual Review of Information Science and Technology*, volume 33. American Society for Information Science, 1998.

[28] Douglas W. Oard, Gina-Anne Levow, and Clara Cabezas. CLEF experiments at the University of Maryland: Statistical stemming and backoff translation strategies. In *Working Notes of the First Cross-Language Evaluation Forum (CLEF-1)*, September 2000. http://www.glue.umd.edu/∼oard/research.html.

[29] Casey Palowitch and Darin Stewart. Automating the structural markup process in the conversion of print documents to electronic text. In *Digital Libraries '95: The Second Annual Conference on the Theory and Practice of Digital Libraries*, Austin, Texas, 1995.

[30] A. Pirkola. The effects of query structure and dictionary setups in dictionary-based cross-language information retrieval. In *Proc. 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 55–63, 1998.

[31] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals Of Speech Recognition*. Prentice Hall, 1993.

[32] P. Resnik, D. W. Oard, and G. Levow. Improved cross-language retrieval using backoff translation. In *Proceedings of the First International Conference on Human Language Technology*, San Diego, 2001. http://www.glue.umd.edu/∼oard/research.html.

[33] Philip Resnik. Mining the web for bilingual text. In *37th Annual Meeting of the Association for Computational Linguistics (ACL'99)*, College Park, MD, 1999.

[34] P. Sibun and A. L. Spitz. Language determination: Natural language processing from scanned document images. In *Proc. 4th Conference on Applied Natural Language Processing*, pages 115–121, Stuttgart, 1994.

[35] A. L. Spitz. Determination of the script and language content of document images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(3):235–245, 1997.

[36] A. Vinciarelli and J. Luettin. Off-line cursive script recognition based on continuous density hmm. In *Proc. 7th International Workshop on Frontiers in Handwriting Recognition*, Amsterdam, 2000.

[37] A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. In *IEEE Transactions on Information Theory*, pages 260–269, 1967.

[38] E. M. Voorhees and D. K. Harman, editors. *The Tenth Text REtrieval Conference (TREC-2001)*, Gaithersburg, MD, 2001. National Institute of Standards and Technology, Department of Commerce. http://trec.nist.gov.

[39] B. Waked, S. Bergler, and C. Y. Suen. Skew detection, page segmentation, and script classification of printed document images. In *IEEE International Conference on Systems, Man, and Cybernetics (SMC'98)*, pages 4470–4475, San Diego, CA, 1998.

[40] R. Weischedel, M. Meteer, R. Schwartz, L. A. Ramshaw, and J. Palmucci. Coping with ambiguity and unknown word through probabilistic models. *Computational Linguistics*, 19(2):359–382, 1993.

[41] G. Jan Wilms. Computerizing a machine readable dictionary. In *Proceedings of the 28th annual Southeast regional conference*, pages 306–313. ACM Press, 1990.

[42] Jinxi Xu, Alexander Fraser, and Ralph Weischedel. Empirical studies in strategies for arabic retrieval. In *25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 269–274, Tampere, Finland, 2002.

[43] Y. Zhu, T. Tan, and Y. Wang. Font recognition based on global texture analysis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(10):1192–1200, 2001.

[44] A. Zramdini and R. Ingold. Optical font recognition using typographical features. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 20(8):877–882, 1998.